NORTH DAKOTA STATE UNIVERSITY

RC Inspection Boat Technical Documentation

SD1212

Kyla Domingo
Scott Larson
Dominic Nelson
Kyle Snyder
4/30/2013

Table of Contents

Project Background **2**

Project Requirements **2**

Pictures Of Our Project **2**

PCB and Enclosures 2

Motors and Drivers 5

Underwater Camera Spool 6

Pontoons and Platform 7

Sonar 7

Assembled Product 8

Schematics **10**

Boat Circuit 10

Remote Circuit 12

Software **13**

XBee/Motor Control-Receiving 13

XBee/Motor Control-Sending 21

GPS 26

Troubleshooting **33**

Boat Circuit 33

Remote Circuit 34

Comments **34**

Boat Circuit 34

Remote Circuit 34

Boat 34

Lessons Learned 34

Appendix **36**

Price List 36

Data Sheets 38

Project Background

There are times where the amount of man power is not adequate enough to help search for victims and/or aquatic vehicles at the bottom of lakes. By creating an entirely wireless inspection boat, we are able to explore the depths of lakes as well as locate missing persons without endangering rescue personnel.

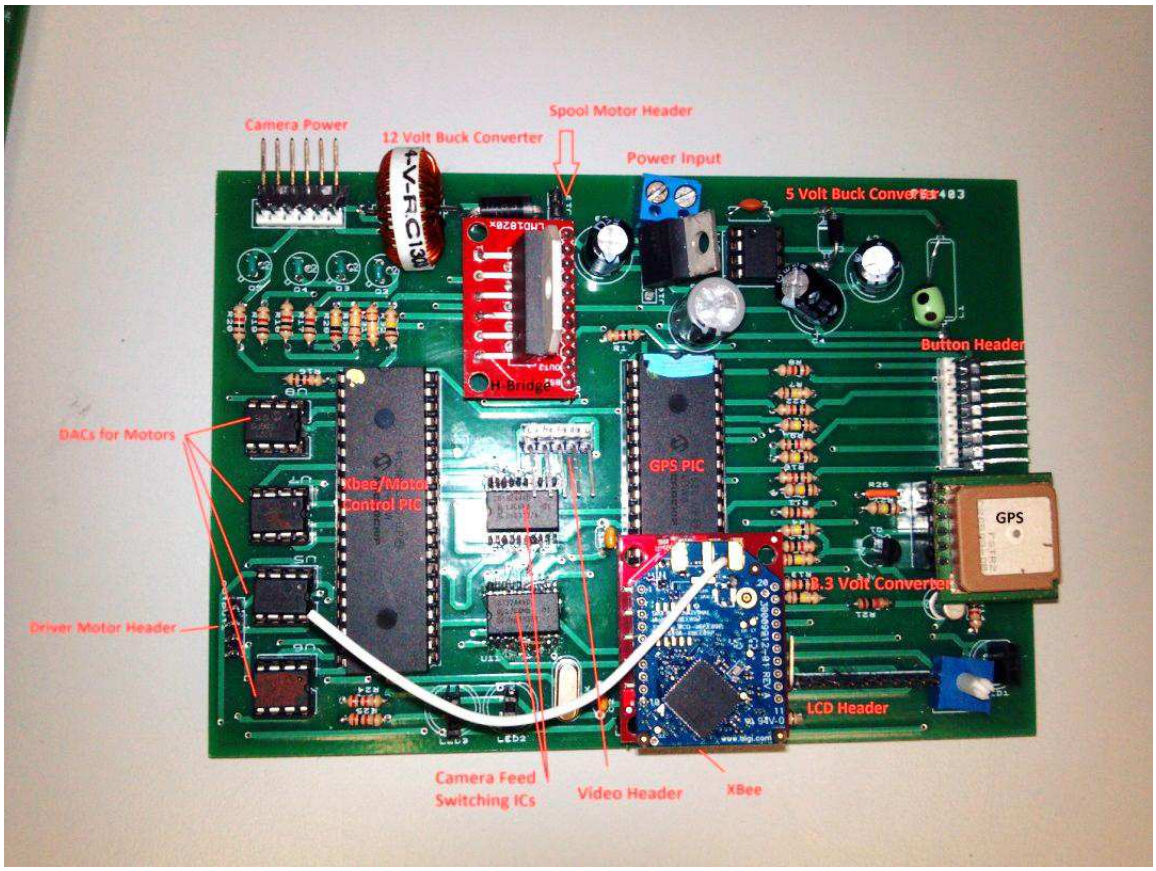
Project Requirements

- The boat should be modular for easy transport
- The boat sections should connect quickly
- All components and materials should be rust resistant and waterproof
- The pontoon will be wirelessly controlled
- Contains one underwater camera
- Contains one forward facing and one reverse facing camera
- Cameras will have at least a 320x200 resolution with a 1 frame per second refresh rate
- Contains sonar for depth and obstacle detection
- Flag staff with an identifying flag as well as a strobe light
- Video feeds should be able to be viewed on standard display
- Video feeds will be toggled on the remote control
- Platform should be approximately 3'x3'
- Pontoons should be approximately 4' long
- Minimum communication range is 1500'
- Steerable and have both forward and reverse capabilities
- Motors to be controlled using two separate joysticks separately
- Battery needs to be rechargeable
- Battery life of at least 4 hours with low power indicator
- Camera should be able to extend at least 60' down
- Must be able to acquire and store four GPS coordinates
- GPS should be toggled on the remote control

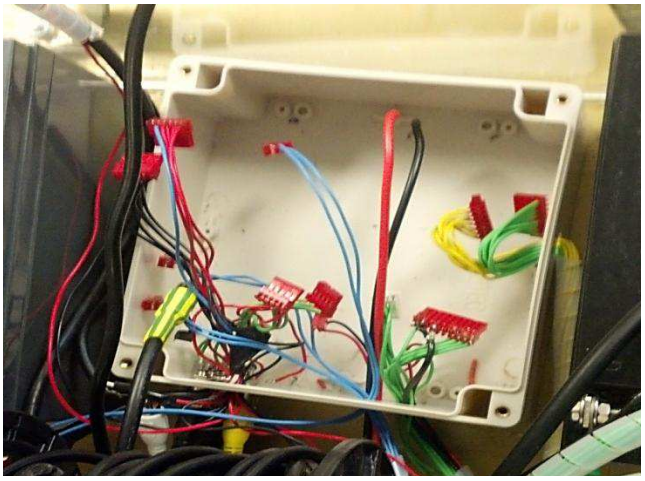
Pictures of the Project

PCB's and Enclosures

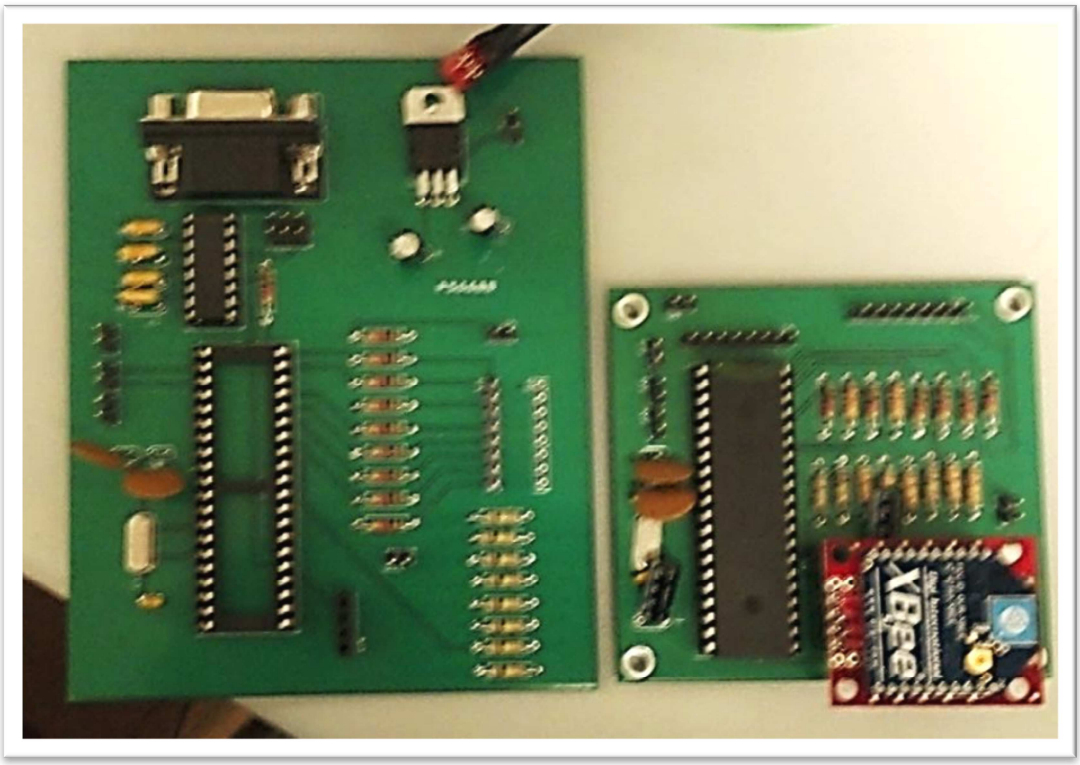
Boat PCB-Handles all the serial data transmitted from the remote



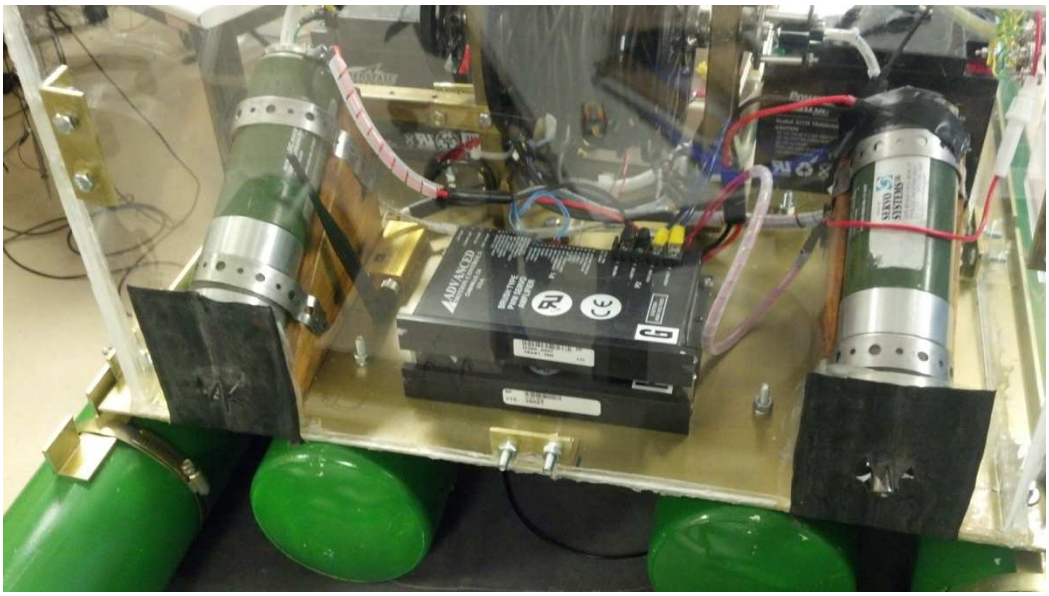
Boat PCB Enclosure- The above PCB is connected to the harnesses below and sealed in the white enclosure below.



Remote PCB-Transmits commands to the boat PCB



Motors and Drivers

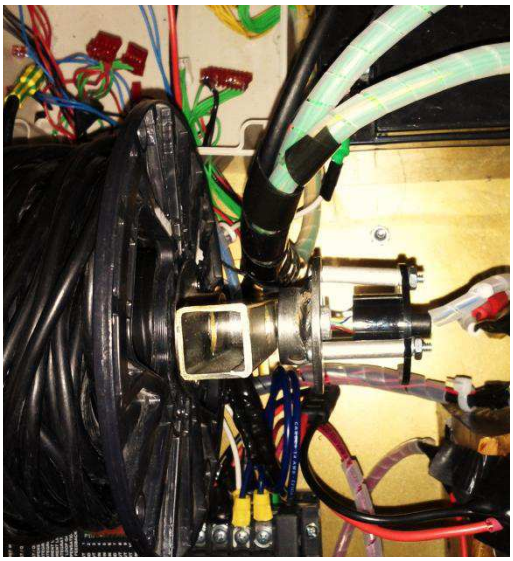


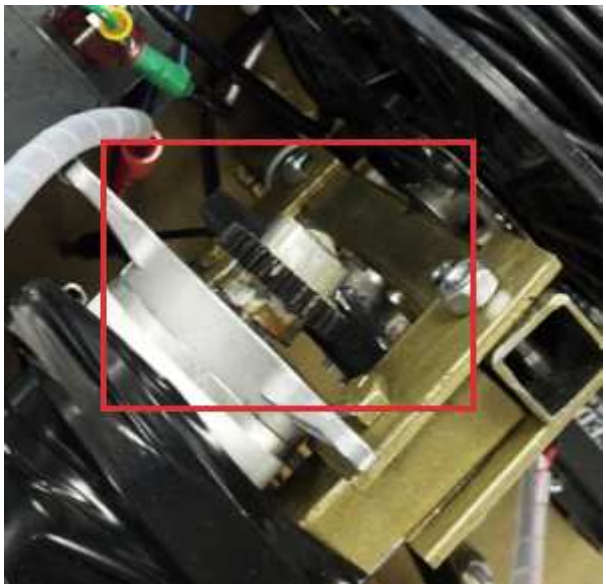
The above picture displays a rear view of the boat and where couplings attach to the propellers.

The picture on the left displays two AMC Servo Amplifiers which are used to drive the two DC motors.

Underwater Camera Spool

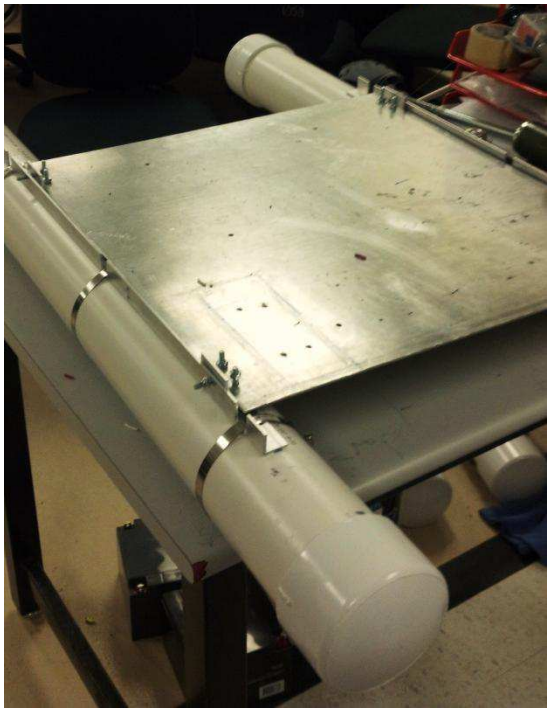
The upper left picture shows the camera spool and the slip ring used to transmit the power and video signals. The upper right picture shows the mechanical connection between the spool and the 12V DC motor used to reel the camera in and out.





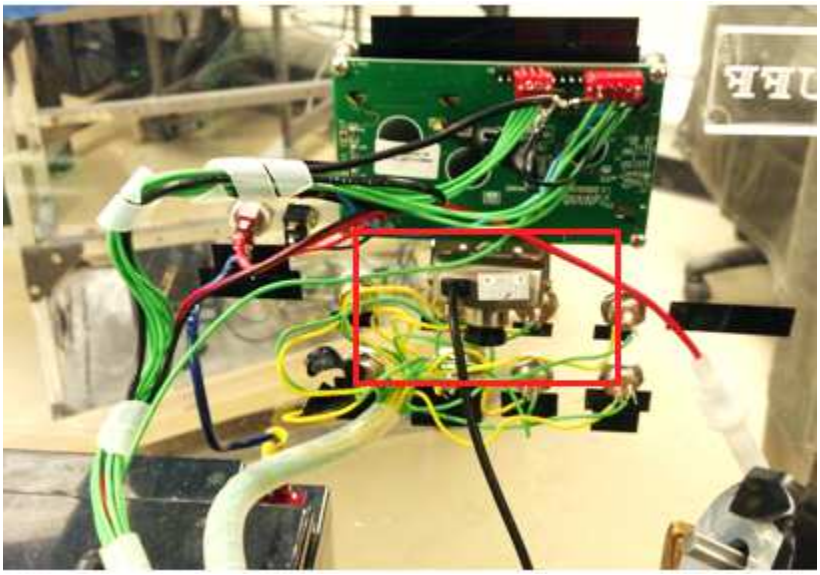
Pontoons and Platform

The below pictures show the platform and pontoons before paint and assembly



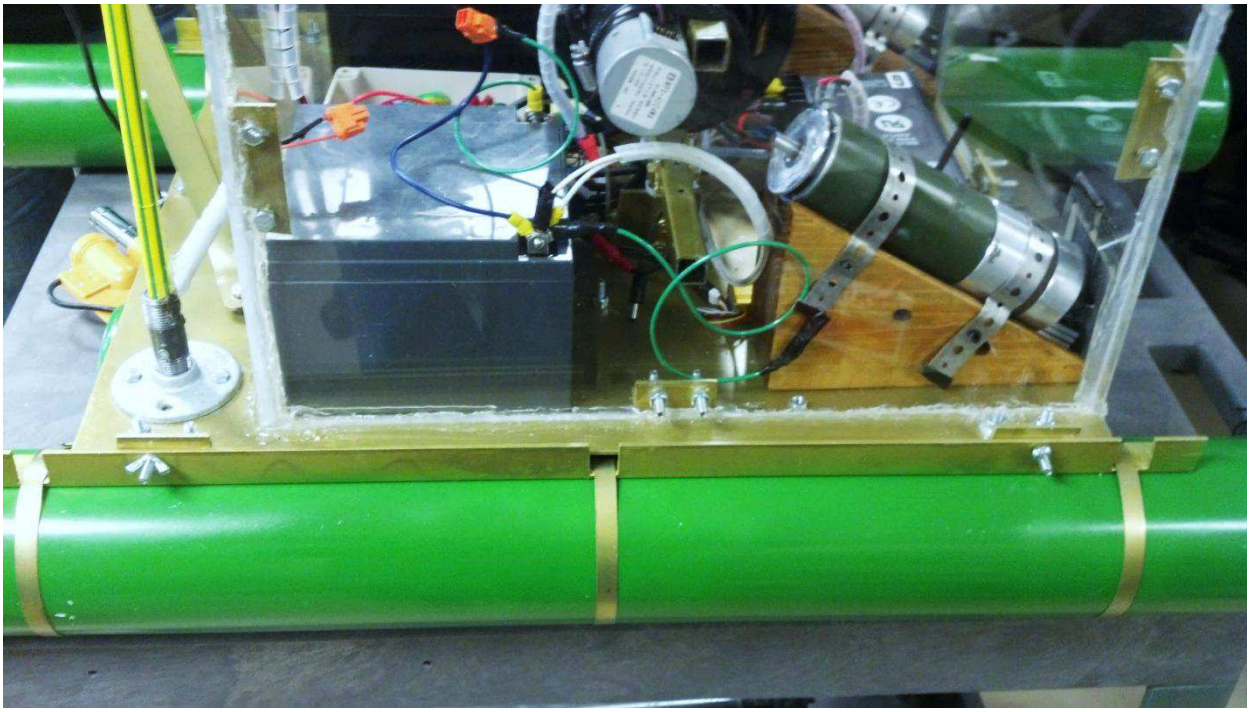
Sonar

Below is a picture of the camera which will point down toward a depth finder (sonar) to show objects of interest below the water.



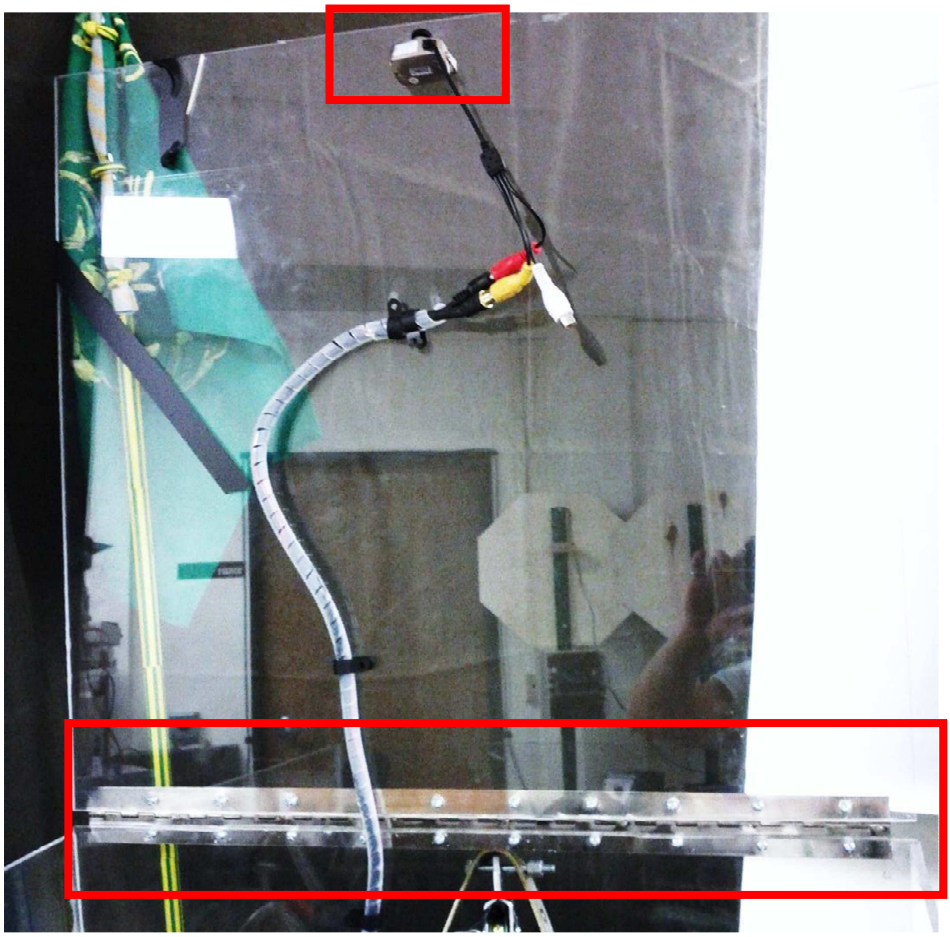
Assembled Product

This photo shows a side view of the boat with the far left being the front of the boat

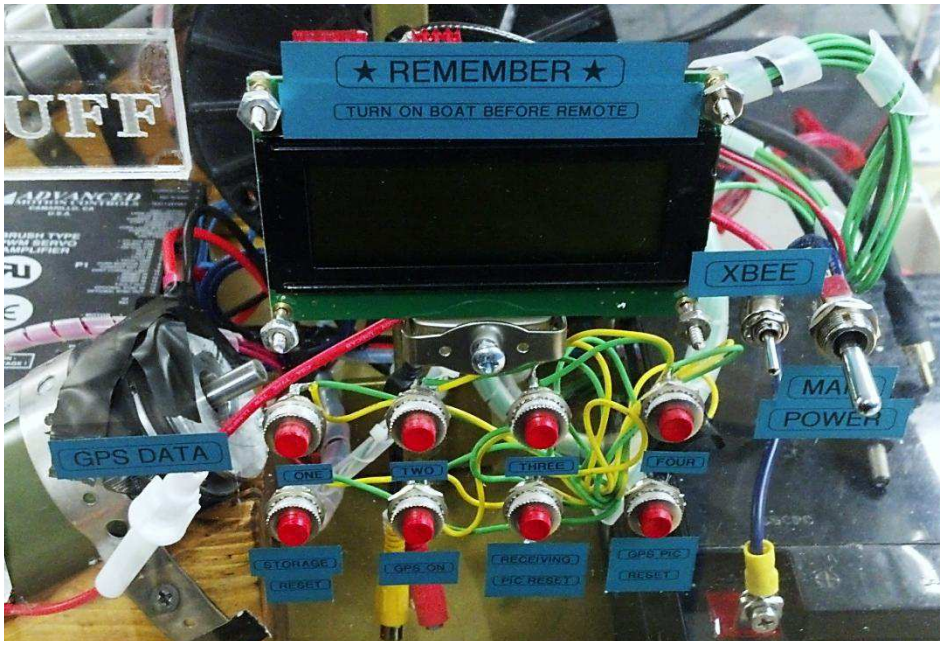


This photo shows a front view of the boat

This photo shows the rear camera and the hinged lid of the boat enclosure.



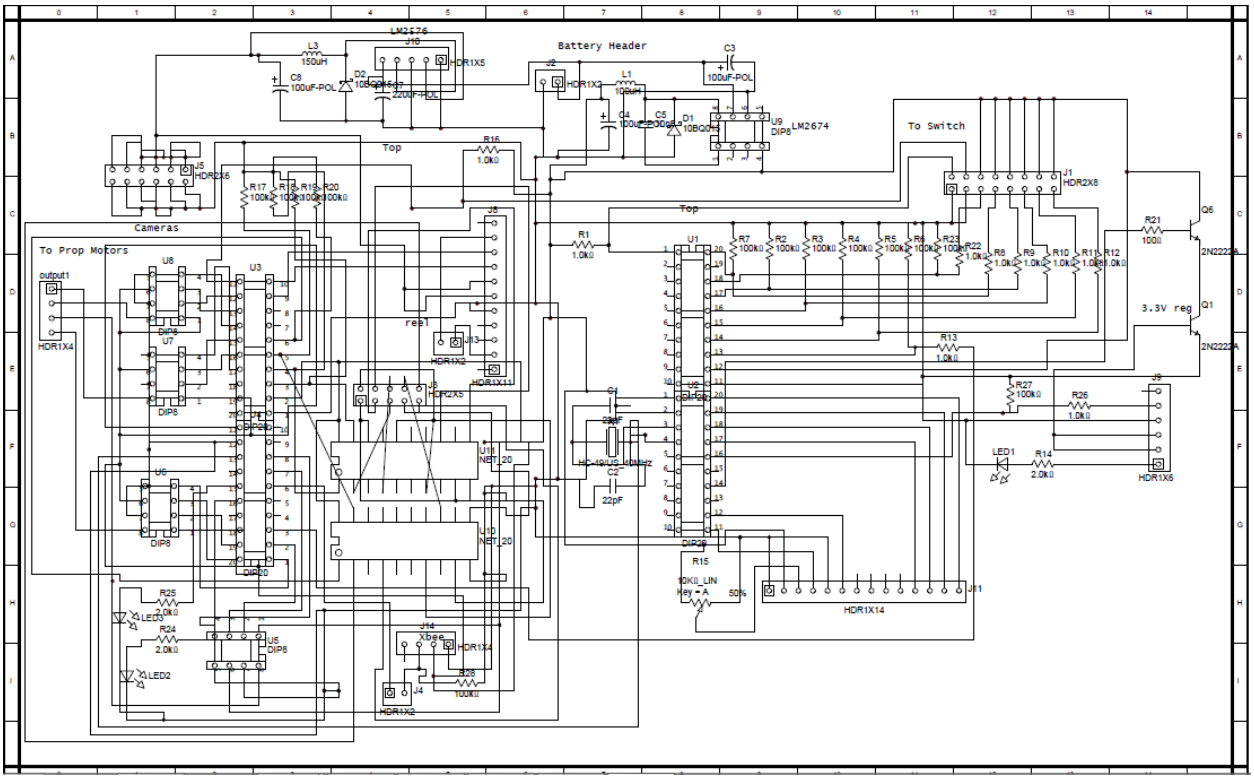
This photo shows the LCD display and the main and Xbee power switches and GPS buttons.



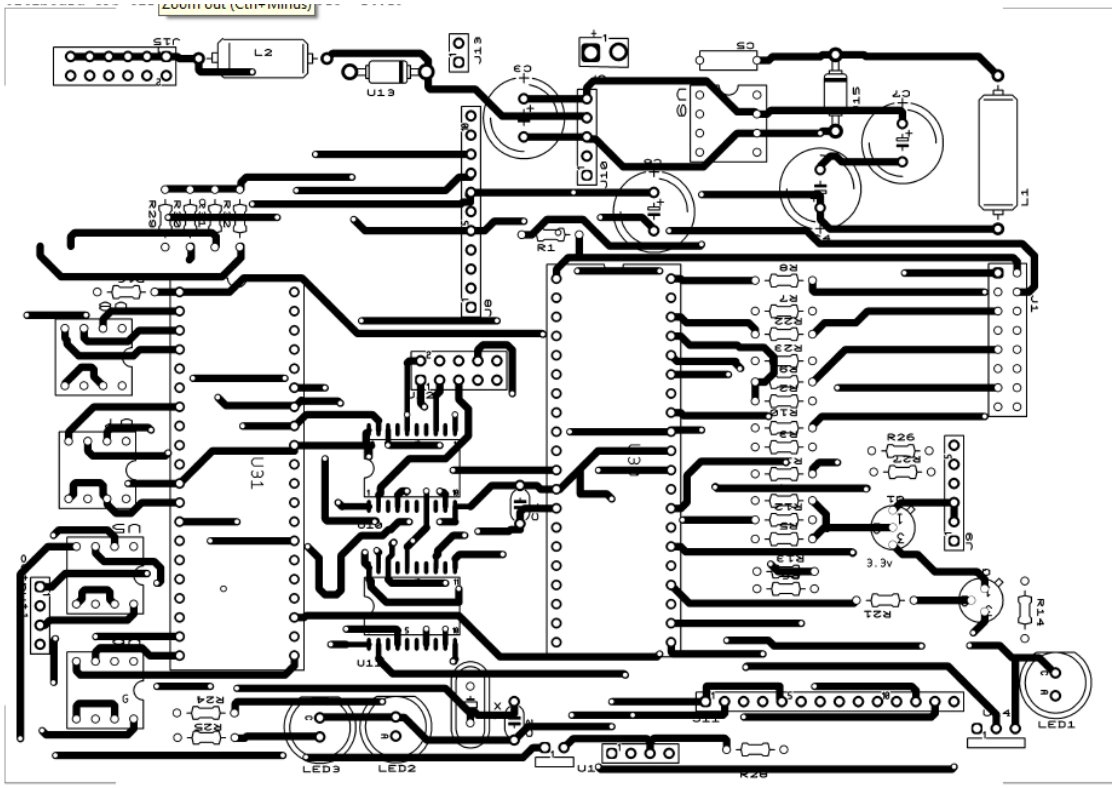
Circuit Schematics

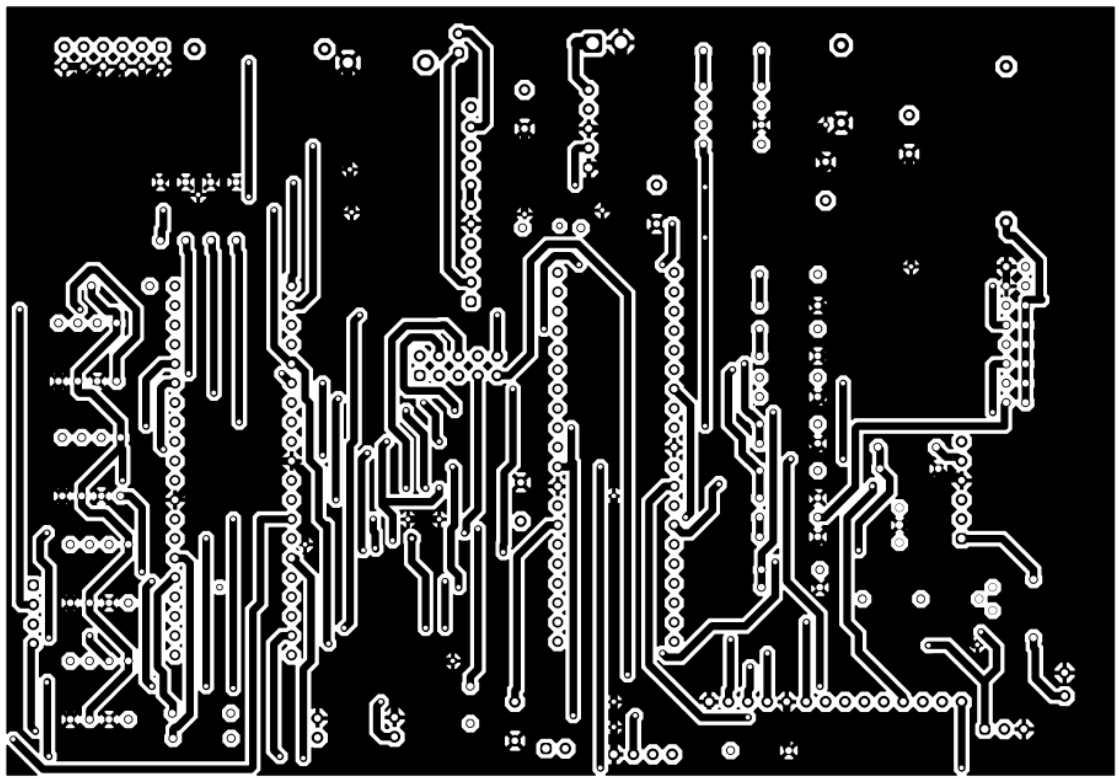
Boat Circuit Schematic

Below is the current version of the circuitry on the boat. The reason two microprocessors was chosen as opposed to one was the amount of serial communication we needed to handle would have created too large of interrupts for one micro to handle. This circuit is responsible for driving everything on the boat, from the driving motors to the cameras. This circuit contains the 12V and 5V power supplies for the components on the boat.

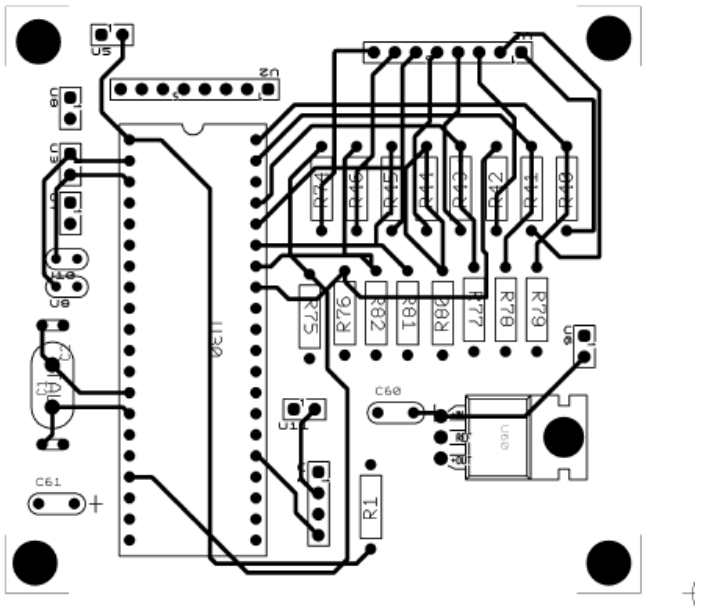


The following are the Cooper Top and Cooper Bottom schematics for the PCB, respectively.



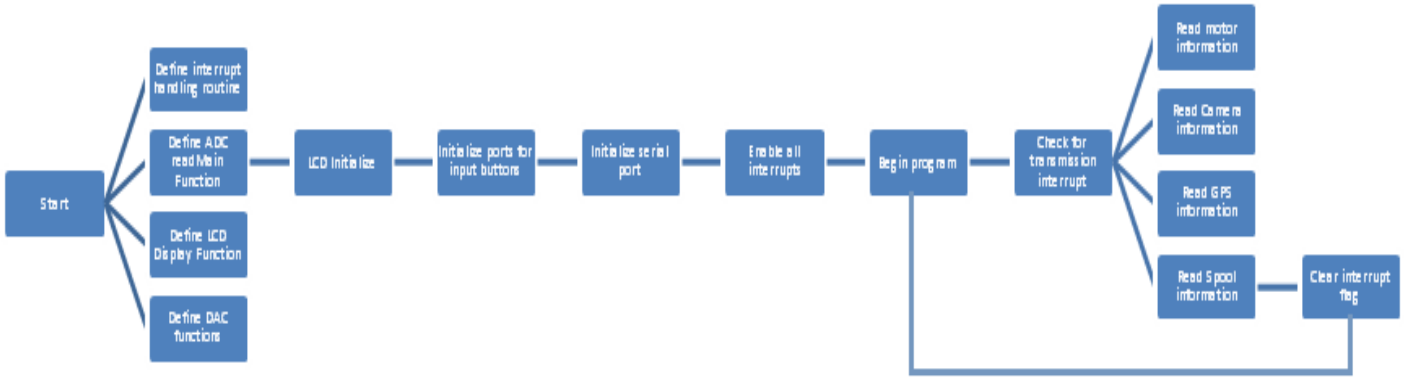


Remote Circuit Schematic



Software

Xbee/Motor Control-Receiving



// Global Variables

```
unsigned char SERIAL1[4];
unsigned char SERIAL2[4];
unsigned char SERIAL3[1];
int F=0;
int J=0;
int K=0;
int L=0;
```

```
unsigned int b;
unsigned char T, i, N, R;
unsigned char MSG[11];
// Subroutine Declarations
#include          <pic18.h>
#include       "function.h"
#include     "math.h"
#include      "lcd_portd.h"
// Subroutines
#include       "function.c"
#include                  "lcd_portd.c"

void interrupt IntServe(void)
{
        if (RCIF) {
                RC0 = !RC0;                                     // debug info for RCIF interrupts
                T = RCREG;
                while(!TRMT); TXREG = T;
                if (T > 32) {
        MSG[N] = T;
                        N += 1 ;
     }
        if (N > 11){ N = 11;}
        if(T ==13){
                if (MSG[0] == 65) {                             // GPRMC message detected
                                for (i=0; i<4; i++) {SERIAL1[i] = (MSG[i+1])-48;}
                                for (i=0; i<4; i++) {SERIAL2[i] = (MSG[i+5])-48;}
                                for (i=0; i<1; i++) {SERIAL3[i] = (MSG[i+9])-48;}
                                F = 1;
                                RC1 = !RC1;                             // debug info - should
toggle 1/sec
                        N = 0;
                }
        }
        RCIF = 0;

        }
}



void D2A(unsigned int X)
                        //Motor one forward
{
        unsigned char i;
        TRISA0 = 0;                                     //CS
        TRISA1 = 0;                                     //SCK
        TRISA2 = 0;                             //SDI
```

```
	X = X & 0x0FFF;
	X = X + 0x3000;
	RA0 = 1;
	RA1 = 1;
	RA0 = 0;

	for (i=0; i<16; i++) // Send out 16 bits of data
		{
			if (X & 0x8000){ RA2 = 1;} else {RA2 = 0;}
			RA1 = 0;
			X = X << 1;
			RA1 = 1;
		}
	RA0 = 1; // CS goes high to terminate the communicaitons
}

void D2A2(unsigned int X)
			//Motor one reverse
{
	unsigned char i;
	TRISA3 = 0;					//CS
	TRISA4 = 0;					//SCK
	TRISA5 = 0;				//SDI
	X = X & 0x0FFF;
	X = X + 0x3000;
	RA3 = 1;
	RA4 = 1;
	RA3 = 0;

	for (i=0; i<16; i++) // Send out 16 bits of data
		{
			if (X & 0x8000) {RA5 = 1;} else {RA5 = 0;}
			RA4 = 0;
			X = X << 1;
			RA4 = 1;
		}
	RA3 = 1; // CS goes high to terminate the communicaitons

}

void D2A3(unsigned int X)
			//Motor one forward
{
	unsigned char i;
	TRISC2 = 0;					//CS
	TRISC3 = 0;					//SCK
	TRISE2 = 0;				//SDI
	X = X & 0x0FFF;
```

```
	X = X + 0x3000;
	RC2 = 1;
	RC3 = 1;
	RC2 = 0;

	for (i=0; i<16; i++) // Send out 16 bits of data
		{
			if (X & 0x8000){ RE2 = 1;} else {RE2 = 0;}
			RC3 = 0;
			X = X << 1;
			RC3 = 1;
		}
	RC2 = 1; // CS goes high to terminate the communicaitons
}

void D2A4(unsigned int X)
			//Motor one reverse
{
	unsigned char i;
	TRISC5 = 0;					//CS
	TRISE0 = 0;					//SCK
	TRISE1 = 0;				//SDI
	X = X & 0x0FFF;
	X = X + 0x3000;
	RC5 = 1;
	RE0 = 1;
	RC5 = 0;

	for (i=0; i<16; i++) // Send out 16 bits of data
		{
			if (X & 0x8000) {RE2 = 1;} else {RE2 = 0;}
			RE0 = 0;
			X = X << 1;
			RE0 = 1;
		}
	RC5 = 1; // CS goes high to terminate the communicaitons
}

unsigned int A2D_Read(unsigned char c)
{
  unsigned int result;
  unsigned char i;
  c = c & 0x0F;
  ADCON0 = (c << 2) + 0x01; // set Channel Select
  for (i=0; i<10; i++); // wait 2.4us (approx)
  GODONE = 1; // start the A/D conversion
  while(GODONE); // wait until complete (approx 8us)
  return(ADRES);
```

```
  }
// Main Routine
void LCD_Out(unsigned int DATA)
{
  unsigned char A[5], i;
  for (i=0; i<5; i++) {
    A[i] = DATA % 10;
    DATA = DATA / 10;
    }
  LCD_Write(ascii(A[4]));
  LCD_Write(ascii(A[3]));
  LCD_Write(ascii(A[2]));
  LCD_Write(ascii(A[1]));
  LCD_Write(ascii(A[0]));
  }

void main(void)
{
  unsigned char i;
  unsigned int A2D;
  unsigned int A2D2;
          //LCD_Init();

          TRISB  = 0;
          ADCON1 = 15;
   TRISC = 0;
          PORTD=0;

          int k=0;
          RB1=1;
          RB2=1;
          RB3=1;
          RB0=1;
          RC4 = 0;
          TRISC = TRISC | 0xC0;              // Initialize Serial Port
          TXIE = 0;
          RCIE = 1;

          BRGH = 0;
          BRG16 = 1;
          SYNC = 0;
          SPBRG = 129;
          TXSTA = 0x22;
          RCSTA = 0x90;
          PEIE = 1;                                    // Turn on all interrupts
          GIE  = 1;
          A2D=500;
          A2D2=500;
```

```
  while(1) {
        if(F==1){

                K=0;
                J=0;
                L=0;
                for(i=0;i<4;i++){K = K + SERIAL1[i]*(pow(10.,i));}
                for(i=0;i<4;i++){J = J + SERIAL2[i]*(pow(10.,i));}
                for(i=0;i<1;i++){L= L+SERIAL3[i];}
                A2D=K;
                A2D2=J;
                b=L;
                F=0;
                }


        if(A2D>540)     {                                                       //Motor one
forward
                                        D2A((A2D-540)*7.1);
                                        D2A2(0*A2D);
                                        }
                else if(A2D<410){
        //Motor one reverse
                                        D2A2(4096-A2D*9.4);
                                        D2A(0*A2D);
                                        }


                else                    {
                                        D2A(0*A2D);
        //Motor one dead
                                        D2A2(0*A2D);
                                        }

                if(A2D2>540)    {
        //Motor two forward
                                        D2A3((A2D2-540)*7.1);
                                        D2A4(0*A2D2);
                                        }
                else if(A2D2<410){
        //Motor two reverse
                                        D2A4(4096-A2D2*9.4);
                                        D2A3(0*A2D2);
                                        }

                else                    {
```

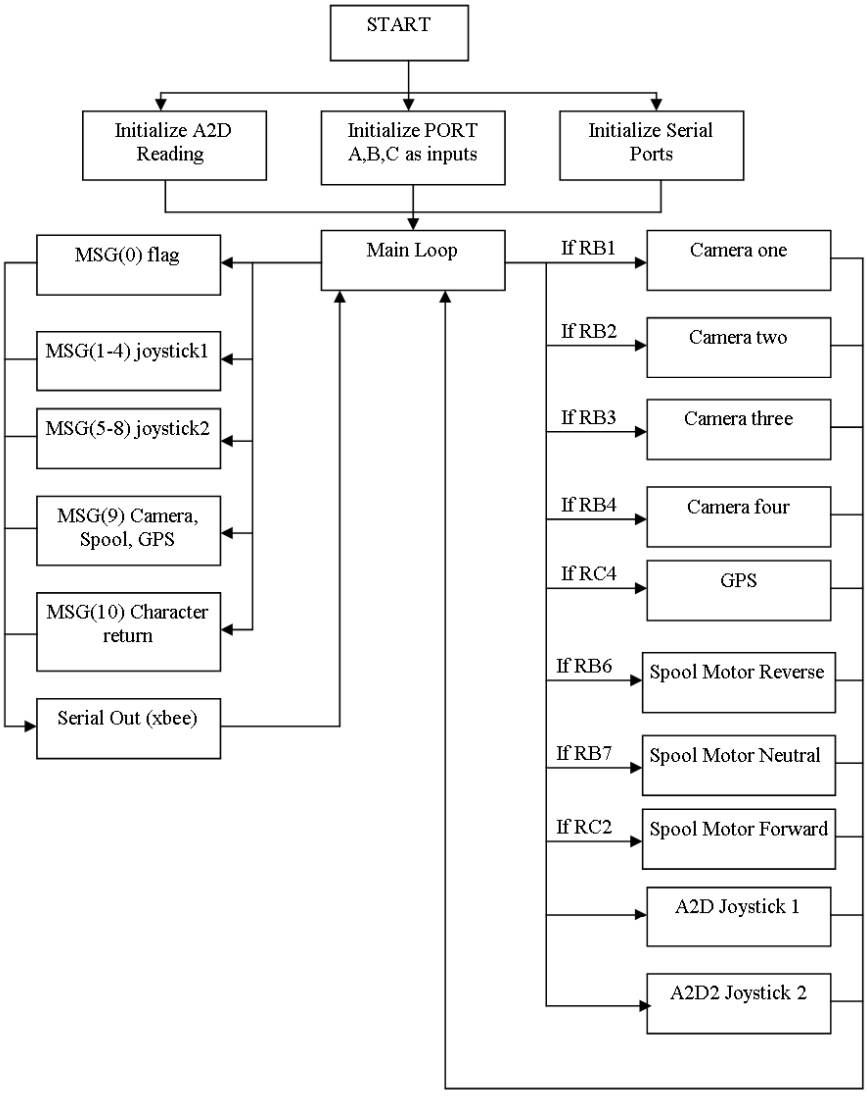
```
                                        D2A3(0*A2D2);
//Motor two dead
                                        D2A4(0*A2D2);
                                        }


        if (b==8)               //camera 1
        {
        RB1=0;
        RB2=1;
        RB3=1;
        RB0=1;
        }
        if(b==1)        //camera 2
        {
        RB1=1;
        RB2=0;
        RB3=1;
        RB0=1;
        }
        if(b==2)        //camera 3
        {
        RB1=1;
        RB2=1;
        RB3=0;
        RB0=1;
        }
        if(b==3)        //camera 4
        {
        RB1=1;
        RB2=1;
        RB3=1;
        RB0=0;
        }
        if(b==7)        //GPS
        {
        RC4=1;
        Wait_ms(5);
        }
        if(b==5)        // spool motor forward
        {
        RB6=1;
        RB7=0;
        }
        if(b==4)        //spool motor reverse
        {
        RB6=0;
```

```
                RB7=1;
                }
                if(b==6)            //dead zone
                {
                RB6=0;
                RB7=0;
                }
                RC4 = 0;
                b=0;

  }
}
```

Xbee/Motor Control-Sending



```
// Global Variables


unsigned char SERIAL[11];                       //the 65 is a flag for the receiving code so that it knows
this data corresponds to motor 1
// Subroutine Declarations
#include          <pic18.h>
#include      "lcd_portd.h"
#include      "function.h"
#include     <math.h>

// Subroutines
#include      "lcd_portd.c"
#include      "function.c"


void A2D_Init(void)
{
  TRISA = 0xFF;
  TRISE = 0x0F;
  ADCON2 = 0x95;
  ADCON1 = 0x07;
  ADCON0 = 0x01;
   }

unsigned int A2D_Read(unsigned char c)
{
  unsigned int result;
  unsigned char i;
  c = c & 0x0F;
  ADCON0 = (c << 2) + 0x01; // set Channel Select
  for (i=0; i<10; i++); // wait 2.4us (approx)
  GODONE = 1; // start the A/D conversion
  while(GODONE); // wait until complete (approx 8us)
  return(ADRES);
}




void LCD_Out(unsigned int DATA)
{
  unsigned char A[5], i;
  for (i=0; i<5; i++) {
    A[i] = DATA % 10;
    DATA = DATA / 10;
    }
  LCD_Write(ascii(A[4]));
```

```
  LCD_Write(ascii(A[3]));
  LCD_Write(ascii(A[2]));
  LCD_Write(ascii(A[1]));
  LCD_Write(ascii(A[0]));
  }


// Main Routine

void main(void)
{
        unsigned int b;
        unsigned char i;
        unsigned int A2D;
        unsigned int VOLT;
        unsigned int CELCIUS;
        unsigned int OHM;
        unsigned int A2D2;
        unsigned int VOLT2;

        TRISA  = 0;                                     //Initializes the ports
        TRISB  = 0xFF;                                  //Initialize port for input from buttons
        TRISD  = 0;
        ADCON1 = 15;
        INT0IE = 1;
        INT0IP = 1;
        TRISB0 = 1;
        INTEDG0 = 1;
        LCD_Init();             // initialize the LCD

        TRISC = TRISC | 0xC0;           // Initialize Serial Port
        TXIE = 0;
        RCIE = 0;
        BRGH = 0;
        BRG16 = 1;
        SYNC = 0;
        SPBRG = 129;

        TXSTA = 0x22;
        RCSTA = 0x90;
//      PEIE = 1;                                       // Turn on all interrupts
        GIE  = 1;
 //
  LCD_Init();           // initialize the LCD
  A2D_Init();
  b=1;                                                  // default to camera 1 on
```

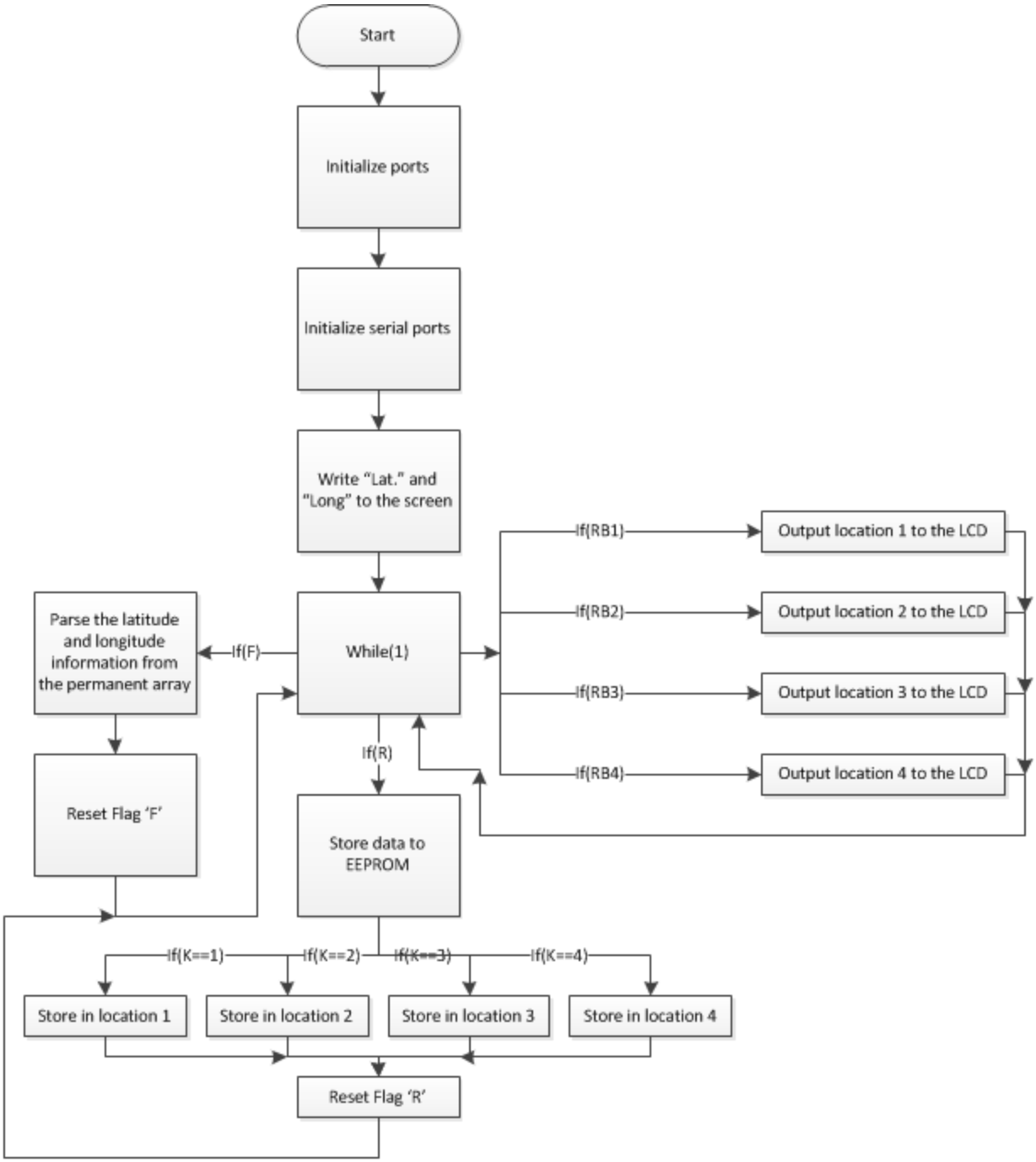
```
        SERIAL[0] = 65;
        SERIAL[10] = 13;
  while(1) {

    A2D = A2D_Read(0);                                                    // Controler
one
         A2D2 = A2D_Read(1);                                              // Controler
two
    LCD_Move(0,10);
         LCD_Out(A2D);
         LCD_Move(1,10);
    LCD_Out(A2D2);
   // Sending Of the Data

        while(RB1==1)
        {
        b=1;
        }
        while(RB2==1)
        {
        b=2;
        }
        while(RB3==1)
        {
        b=3;
        }
        while(RB4==1)
        {
        b=4;
        }
        while(RB5==1)
        {
        b=5;
        }
        while(RB6==1)
        {
        b=6;
        }
        while(RB7==1)
        {
        b=7;
        }
        while(RC2==1)
        {
        b=8;
        }
```

```
        for(i=1;i<5;i++){
                SERIAL[i] = (A2D % 10)+48;
                A2D = A2D/10;
        }
        for(i=5;i<9;i++){
                SERIAL[i] = (A2D2 % 10)+48;
                A2D2 = A2D2/10;
        }
        for(i=9;i<10;i++){
                SERIAL[i]=b+48;
                }
        for(i=0;i<11;i++){
                while(!TRMT); TXREG = SERIAL[i];
        }
        b=0;    // to clear b
 }
}
```

GPS




```
// Global Variables
const unsigned char MSG0[5] = "PT. 1";
const unsigned char MSG3[5] = "PT. 2";
const unsigned char MSG4[5] = "PT. 3";
const unsigned char MSG5[5] = "PT. 4";
const unsigned char MSG1[4] = "LAT:";
```

```
const unsigned char MSG2[5] = "LONG:";
const unsigned char MSG6[2] = " ";
unsigned char MSG[81]; // last four messages
unsigned char GPS[81]; // last four messages
unsigned char T, i, N, R;
unsigned int K=0;
unsigned int Q,W,E,G;
unsigned char F,LONGITUDE1, LATITUDE1, LONGITUDE2, LATITUDE2, LONGITUDE3, LATITUDE3,
LATITUDE4, LONGITUDE4;

// Subroutine Declarations
#include          <pic18.h>

// Subroutines
#include      "function.c"
#include      "lcd_portd.c"

// Interrupt service routine
void interrupt IntServe(void)
{
        if (RCIF) {
                RC0 = !RC0;                                     // debug info for RCIF interrupts
                T = RCREG;
                if (T > 32) {
        MSG[N] = T;
                        N += 1 ;
    }
        if (N > 80){ N = 80;}
        if (T == 13) {
                if (MSG[3] == 'R') {                            // GPRMC message detected
                        for (i=0; i<80; i++) {GPS[i] = MSG[i];}
                        F = 1;
                        RC1 = !RC1;                             // debug info - should toggle
1/sec
                        }
                N = 0;
                }
        RCIF = 0;
        }
        if (INT0IF){                                            // Looks for a button press to store the
current location
```

```
                K++;                                                    // Increments the variable that
corresponds to the location number
                R=1;                                                    // Sets the flag for writing the
data to the EEPROM
                RC2=!RC2;
                INT0IF = 0;

        }
}
void LCD_Out(int DATA)                                          // Writes information to the LCD
{
        int A[3], i;
        for (i=0; i<3; i++) {
                        A[i] = DATA % 10;
                        DATA = DATA / 10;
                }

        for (i=3; i>0; i--){
                LCD_Write(A[i-1] + '0');
        }
}
// Main Routine

void main(void)
{
        K=0;                                                    //Resets the variable for the storage loctaion
        TRISA  = 0;                                             //Initializes the ports
        TRISB  = 0xFF;
        TRISD  = 0;
        TRISC  = 0;
        ADCON1 = 15;
        INT0IE = 1;
        INT0IP = 1;
        TRISB0 = 1;
        INTEDG0 = 1;
        LCD_Init();
        LCD_Init();             // initialize the LCD
        RC5 = 0;
        TRISC = TRISC | 0xC0;           // Initialize Serial Port
        TXIE = 0;
        RCIE = 1;
        BRGH = 0;
```

```
        BRG16 = 1;
        SYNC = 0;
        SPBRG = 129;
        TXSTA = 0x22;
        RCSTA = 0x90;
        PEIE = 1;                                       // Turn on all interrupts
        GIE  = 1;

        LCD_Move(1,0); for(i=0;i<4;i++) LCD_Write(MSG1[i]);     // writes the messege "Lat." to the LCD.
        LCD_Move(2,0); for(i=0;i<5;i++) LCD_Write(MSG2[i]);     // writes the messege "Long" to the
LCD.
   while(1) {
                while(RB5){
                RC5 =1;
                }
                while(RB7) K=0;
        // Alows the user to reset the location variable
                if(F) {
        // Stores the relevent GPS data into individual variables
                        LATITUDE1 = (GPS[20]-'0')*10 + (GPS[21]-'0');
                        LATITUDE2 = (GPS[22]-'0')*10 + (GPS[23]-'0');
                        LATITUDE3 = (GPS[25]-'0')*10 + (GPS[26]-'0');
                        LATITUDE4 = (GPS[27]-'0');
                        LONGITUDE1 = (GPS[32]-'0')*100 + (GPS[33]-'0')*10 + (GPS[34]-'0');
                        LONGITUDE2 = (GPS[35]-'0')*10 + (GPS[36]-'0');
                        LONGITUDE3 = (GPS[38]-'0')*10 + (GPS[39]-'0');
                        LONGITUDE4 = (GPS[40]-'0');
                        RC3=!RC3;
                        F=0;
        //Resets the flag
                }
                if(R){
                        if(K==1){
        //Stores the 1st location in memory
                                EEPROM_WRITE(0x00, LATITUDE1);
                                EEPROM_WRITE(0x01, LATITUDE2);
                                EEPROM_WRITE(0x02, LATITUDE3);
                                EEPROM_WRITE(0x09, LATITUDE4);
                                EEPROM_WRITE(0x10, LONGITUDE1);
                                EEPROM_WRITE(0x11, LONGITUDE2);
                                EEPROM_WRITE(0x12, LONGITUDE3);
                                EEPROM_WRITE(0x19, LONGITUDE4);
```

```
                    }
                if(K==2){
        //Stores the 2nd location in memory
                    EEPROM_WRITE(0x03, LATITUDE1);
                    EEPROM_WRITE(0x04, LATITUDE2);
                    EEPROM_WRITE(0x05, LATITUDE3);
                    EEPROM_WRITE(0x26, LATITUDE4);
                    EEPROM_WRITE(0x13, LONGITUDE1);
                    EEPROM_WRITE(0x14, LONGITUDE2);
                    EEPROM_WRITE(0x15, LONGITUDE3);
                    EEPROM_WRITE(0x27, LONGITUDE4);
                    }
                if(K==3){
        //Stores the 3rd location in memory
                    EEPROM_WRITE(0x06, LATITUDE1);
                    EEPROM_WRITE(0x07, LATITUDE2);
                    EEPROM_WRITE(0x08, LATITUDE3);
                    EEPROM_WRITE(0x28, LATITUDE4);
                    EEPROM_WRITE(0x16, LONGITUDE1);
                    EEPROM_WRITE(0x17, LONGITUDE2);
                    EEPROM_WRITE(0x18, LONGITUDE3);
                    EEPROM_WRITE(0x29, LONGITUDE4);
                    }
                if(K==4){
        //Stores the 4th location in memory
                    EEPROM_WRITE(0x20,LATITUDE1);
                    EEPROM_WRITE(0x21, LATITUDE2);
                    EEPROM_WRITE(0x22, LATITUDE3);
                    EEPROM_WRITE(0x30, LATITUDE4);
                    EEPROM_WRITE(0x23, LONGITUDE1);
                    EEPROM_WRITE(0x24, LONGITUDE2);
                    EEPROM_WRITE(0x25, LONGITUDE3);
                    EEPROM_WRITE(0x31, LONGITUDE4);
                    }
                R=0;
        // Resets the flag
            }
        if(RB1){                        //Writes the data corresponding to the 1st location to
the LCD
            LCD_Move(0,0);
            for(i=0;i<5;i++) LCD_Write(MSG0[i]);
            LCD_Move(1,5);
```

```
            Q = EEPROM_READ(0x02);
            E = Q*10+EEPROM_READ(0x09);
            W = EEPROM_READ(0x12);
            G = W*10+EEPROM_READ(0x19);
            LCD_Out(EEPROM_READ(0x00));LCD_Write(42);LCD_Write(32);
            LCD_Out(EEPROM_READ(0x01));LCD_Write('.');LCD_Out(E);LCD_Write(39);
            LCD_Move(2,6);
            LCD_Out(EEPROM_READ(0x10));LCD_Write(42);LCD_Write(32);
            LCD_Out(EEPROM_READ(0x11));LCD_Write('.');LCD_Out(G);LCD_Write(39);
        }
        if(RB2){                                //Writes the data corresponding to the 2nd location to
the LCD
            LCD_Move(0,0);
            for(i=0;i<5;i++) LCD_Write(MSG3[i]);
            LCD_Move(1,5);
            Q = EEPROM_READ(0x05);
            E = Q*10+EEPROM_READ(0x26);
            W = EEPROM_READ(0x15);
            G = W*10+EEPROM_READ(0x27);
            LCD_Out(EEPROM_READ(0x03));LCD_Write(42);LCD_Write(32);
            LCD_Out(EEPROM_READ(0x04));LCD_Write('.');LCD_Out(E);LCD_Write(39);
            LCD_Move(2,6);
            LCD_Out(EEPROM_READ(0x13));LCD_Write(42);LCD_Write(32);
            LCD_Out(EEPROM_READ(0x14));LCD_Write('.');LCD_Out(G);LCD_Write(39);
        }
        if(RB3){                                //Writes the data corresponding to the 3rd location to
the LCD
            LCD_Move(0,0);
            for(i=0;i<5;i++) LCD_Write(MSG4[i]);
            LCD_Move(1,5);
            Q = EEPROM_READ(0x08);
            E = Q*10+EEPROM_READ(0x28);
            W = EEPROM_READ(0x18);
            G = W*10+EEPROM_READ(0x29);
            LCD_Out(EEPROM_READ(0x06));LCD_Write(42);LCD_Write(32);
            LCD_Out(EEPROM_READ(0x07));LCD_Write('.');LCD_Out(E);LCD_Write(39);
            LCD_Move(2,6);
            LCD_Out(EEPROM_READ(0x16));LCD_Write(42);LCD_Write(32);
            LCD_Out(EEPROM_READ(0x17));LCD_Write('.');LCD_Out(G);LCD_Write(39);
        }
        if(RB4){                                //Writes the data corresponding to the 4th location to
the LCD
```

```
                LCD_Move(0,0);
                for(i=0;i<5;i++) LCD_Write(MSG5[i]);
                Q = EEPROM_READ(0x22);
                E = Q*10+EEPROM_READ(0x30);
                W = EEPROM_READ(0x25);
                G = W*10+EEPROM_READ(0x31);
                LCD_Move(1,5);
                LCD_Out(EEPROM_READ(0x20));LCD_Write(42);LCD_Write(32);
                LCD_Out(EEPROM_READ(0x21));LCD_Write('.');LCD_Out(E);LCD_Write(39);
                LCD_Move(2,6);
                LCD_Out(EEPROM_READ(0x23));LCD_Write(42);LCD_Write(32);
                LCD_Out(EEPROM_READ(0x24));LCD_Write('.');LCD_Out(G);LCD_Write(39);
        }
 }
}
```

Troubleshooting

***Warning: Any troubleshooting done on the boat or the remote should be done with all batteries disconnected to avoid shock.**

Boat Circuit

Problem	Causes	Solution
Boat is not powered	1. Battery is not charged. 2. Loose connection between the two batteries. 3. Voltage regulator is disconnected or burnt.	1. Disconnect battery and charge to 12V. 2. Check the connections between the batteries and from the batteries to the PCB enclosure. 3. Replace voltage regulator
Xbee not receiving	1. Xbee is not powered. 2. PIC code is reset.	1. Check connection to Xbee and make sure Xbee switch is in the “ON” position. 2. Reprogram the receiving PIC.
Camera feed doesn’t work	1. Camera power or video cable is unplugged. 2. Camera switching IC is burnt out. 3. Button is stuck on the remote.	1. Check connections for camera power and video feed. 2. Replace the switching IC. 3. Make sure button is fully released.
GPS does not mark coordinates	1. GPS is not being powered. 2. GPS is not connected to satellites.	1. Check power connection PCB enclosure. 2. Move to open location and wait 2 minutes for the GPS to locate satellites.
Motors will not spin	1. Motor is not connected. 2. Xbee is not transmitting.	1. Bad connection between servo amplifier and the motor. 2. Review above solution for Xbee transmission.
Spool spins on startup	1. Motor is connected before PIC is started.	1. Turn power to the boat off and disconnect the motor.
Flag LED not flashing	1. Power is not connected	1. Connect power to LED
LCD Display freezes	1. Interference from on board components.	1. Shield cable disconnected from ground. 2. Shielding capacitor is disconnected.

Remote Circuit

Remote is not powered	1. Battery is dead. 2. Connection between battery and PCB is disconnected.	1. Replace 9V battery. 2. Disassemble remote and check for loose wire.
Remote is not transmitting	1. Xbee is not powered. 2. Intermittent connection issue. 3. Code is cleared on PIC	1. See above step. 2. Check PCB for bad connection. 3. Reprogram PIC
Buttons working incorrectly	1. Loose connection. 2. Button is stuck.	1. Check the terminals of the button for loose wires. 2. Make sure button is fully released.
Joysticks do not respond	1. Loose connection.	1. Check connection to joystick.

Comments

Boat Circuit

Some of the that could be made to the boat are an improved H-bridge circuit that would allow for better control speed for the spool. Anther improvement would be more reliable camera IC's that would allow for the camera to switch on and off.

Remote Circuit

Some improvements that could be made to the remote are the addition of a LCD display screen that would display the GPS locations. Another improvement would a low battery indicator for the remote and the boat that would be displayed on the remote enclosure. Another improvement would be a more user friendly enclosure and placement of joysticks.

Boat

Improvements that could have been made to the boat include:

- Bigger platform
- Larger pontoons
- Separate spool enclosure
- More adaptable system for motor mounting
- More water tight enclosure

Lessons Learned

- When using long wire leads are used, shielded wire, twisted pairs, and/or discharge capacitor are necessary to reduce noise.
- Good communication between group members made the project more successful
- Earlier completion date allowed time for testing and trouble shooting
- Create an organized plan for testing-testing and retesting the same idea is not helpful.
- Create copies of all documents

    - Code
    - Ordered parts
    - Documents
- Price Variations
- GPS logic is only 3.3V, which cannot be directly read by the PIC
- Accounting for shipping lead times
- Accounting for ordering lead times

Appendix

Product	Number	Price	Total	Comment
MCP 4921	4	$2.36	$9.44	DAC
PIC18F4620	3	$7.94	$23.82	Micro
CBT3244AD	2	$0.52	$1.04	Camera Switching
LMD18200T	1	$16.04	$16.04	H-Bridge
LM7805C	1	$0.67	$0.67	5V linear regulator
LM2576T	1	$2.83	$2.83	12V linear regulator
LP2950	1	$0.47	$0.47	3.3 V regulator
LM2674	1	$3.25	$3.25	5 V regulator
Xbee Pro	2	$66.95	$133.90	xbee
COM-09032	2	$3.95	$7.90	Joysticks
916-UP501	1	$31.35	$31.35	GPS
DC Motor	2	$62.32	$124.64	
Servo motor	1	$28.50	$28.50	
30A8t	2	$50.00	$100.00	
12V Battery	2	$50.00	$100.00	
Aluminum Platform	1	$200.00	$200.00	
Aluminum angle iron	2	$11.99	$23.98	
Cameras	3	$8.83	$26.50	
Underwater Camera	1	$75.00	$75.00	
Buttons	8	$2.95	$23.60	
Plexi glass	1	$113.96	$113.96	
LCD	1	$25.90	$25.90	
Flat Iron	1	$9.89	$9.89	
Props	2	$13.99	$27.98	
Shaft Arbor	2	$7.68	$15.36	
Shaft Coupling	2	$12.02	$24.04	
Hose Clamp	10	$1.89	$18.90	
Breakout Board	2	$1.95	$3.90	
Motor Shaft	1	$8.99	$8.99	
PVC Pipe	4	$7.72	$30.88	
PVC Caps	8	$5.38	$43.04	
Remote Enclosure	1	$16.59	$16.59	
Boat Enclosure	1	$16.23	$16.23	
Transmitter/Reciever	1	$59.99	$59.99	
Bolts	46	$0.37	$17.02	
Pulley	1	$5.29	$5.29	

Nuts	42	$0.40	$16.80	
Wing Nuts	4	$0.10	$0.40	
Slip Ring	1	$17.50	$17.50	
PCB	2	$29.99	$59.98	
Paint	2	$7.99	$15.98	
Lock washers	16	$0.18	$2.88	
Flag	1	$2.49	$2.49	
PVC Cement	1	$2.29	$2.29	
Misc from tool room	1	$75.00	$75.00	
			$1,564.21	

Datasheets saved on CD.